

CSE 259 - Logic in Computer Science

Fall 2024

Recitation-10

Waqar Hassan Khan



Project 3

- The partial implementation **wangs_algorithm.pl** is runnable
- Sample input/output:
 - Premises as list ($[p \wedge q]$)
 - A formula as conclusion
 - We can derive the conclusion from the premise

```
| ?- run([p^q], [p]).  
[p^q]=>[p]  
=      [p,q]=>[p]      (by and/left)  
=      Done (sharing a/f)
```

We can derive the conclusion from the premises.

yes

Project 3

- Write your codes here
- I have implemented some. The rest is your task

```
%%%%%%%%%% YOUR CODE STARTS %%%%%%%%%%  
  
% Implement all other non-branching rules below by following Wangs algorithm  
  
> /* ...  
> prove(L => R):- ...  
> prove(L => R):- ...  
> /* ...  
> prove(L => R) :- ...  
> prove(L => R) :- ...  
% Implement all branching rules below by following Wangs algorithm  
> /* ...  
> prove(L => R) :- ...  
%%%%%%%%% YOUR CODE ENDS %%%%%%%%%%
```

Project 3

- Two types of rules
 - Non-branching
 - Branching

Project 3: Non-branching Rules

Rule-1

If one of the formulae separated by commas is the negation of a formula, drop the negation sign and move it to the other side of the arrow.

Example:

Formula: $p, \sim(q \wedge r) \Rightarrow p \wedge r$

Change to: $p \Rightarrow p \wedge r, q \wedge r$

Project 3: Non-branching Rules

Rule-2

If the last connective of a formula on the left is \wedge (and), or on the right of the arrow is \vee (or), replace the connective by a comma.

Example-1:

Formula: $p, p \wedge q \Rightarrow r, s$

Change to: $p, p, q \Rightarrow r, s$

Example-2:

Formula: $p, q \Rightarrow r \vee p$

Change to: $p, q \Rightarrow r, p$

Project 3: Non-branching Rules

Rule-3

If the last connective of a formula on the right is $A \rightarrow B$, remove $A \rightarrow B$ from the right and then add A to the left and B to the right.

.

Example-1:

Formula: $p \vee q \Rightarrow q \rightarrow p$

Change to: $p \vee q, q \Rightarrow p$

Project 3: Branching Rules

Rule-4

If the last connective of a formula on the left is \vee (or), or on the right of the arrow is \wedge (and), then produce two new lines, each with one of the two sub formulae replacing the formula. Both of these must be proved in order to prove the original theorem.

Example-1:

Formula: $p \vee q, r, s \Rightarrow q \vee p$

Change to:

$p, r, s \Rightarrow q \vee p$

$q, r, s \Rightarrow q \vee p$

Example-2:

Formula: $p, r, s \Rightarrow q \wedge p$

Change to:

$p, r, s \Rightarrow q$

$p, r, s \Rightarrow p$

Project 3: Branching Rules

Rule-5

If the last connective of a formula on the left is $A \rightarrow B$, remove $A \rightarrow B$ from the left and then create two new lines, one with B added to the left, and the other with A added to the right.

Example-1:

Formula: $p, p \rightarrow q \Rightarrow p \vee q$

Change to:

$p, q \Rightarrow p \vee q$

$p \Rightarrow p \vee q, p$

Project 3: Non-branching Rules – Rule 1

```
/*
 * Rule-1
 * example rule: negation
 * non-branching rule
 * If one of the formulae separated by commas is the
 * negation of a formula, drop the negation sign and
 * move it to the other side of the arrow.
 */
prove(L => R):-
    member(~X, L),
    del(~X, L, NewL),
    nl, write('=\\t'), write(NewL => [X | R]),
    write('\\t (by negation/left)'),
    prove(NewL => [X | R]).
prove(L => R):-
    member(~X, R),
    del(~X, R, NewR),
    nl, write('=\\t'), write([X | L] => NewR),
    write('\\t (by negation/right)'),
    prove([X | L] => NewR).
```

Example:

Formula: $p, \sim(q \wedge r) \Rightarrow p \wedge r$

Change to: $p \Rightarrow p \wedge r$

Output:

```
| ?- prove([p, ~(q ^ r)] => [p ^ r]).
=          [p] => [q ^ r, p ^ r]    (by negation/left)
```

Project 3: Non-branching Rules – Rule 2

```
/*
 * Rule-2
 * example rule: left conjunction
 * non-branching rule
 * If the last connective of a formula on the left is  $\wedge$  (and),
 * or on the right of the arrow is  $\vee$  (or), replace the connective by a comma.
 */
prove(L  $\Rightarrow$  R) :-
  member(A  $\wedge$  B, L),
  del(A  $\wedge$  B, L, NewL),
  nl, write('= \t'), write([A, B | NewL]  $\Rightarrow$  R),
  write(' \t (by and/left)'),
  prove([A, B | NewL]  $\Rightarrow$  R).
prove(L  $\Rightarrow$  R) :-
  member(A  $\vee$  B, R),
  del(A  $\vee$  B, R, NewR),
  nl, write('= \t'), write(L  $\Rightarrow$  [A, B | NewR]),
  write(' \t (by or/right)'),
  prove(L  $\Rightarrow$  [A, B | NewR]).
```

Project 3: Non-branching Rules – Rule 2 contd.

Example-1:

Formula: $p, p \wedge q \Rightarrow r, s$

Change to: $p, p, q \Rightarrow r, s$

```
--  
| ?- prove([p, p ^ q] => [r, s]).  
  
=      [p,q,p]=>[r,s]    (by and/left)
```

Example-2:

Formula: $p, q \Rightarrow r \vee p$

Change to: $p, q \Rightarrow r, p$

```
--  
| ?- prove([p, q] => [r v p]).  
  
=      [p,q]=>[r,p]      (by or/right)  
=      Done (sharing a/f)
```

Project 3: Branching Rules – Rule 5

```
/*  
 * Rule-5  
 * example rule: left implication  
 * branching rule  
 * If the last connective of a formula on the left is  $A \rightarrow B$ ,  
 * remove  $A \rightarrow B$  from the left and then create two new lines,  
 * one with  $B$  added to the left, and the other with  $A$  added to the right.  
 */
```

```
prove(L => R) :-  
  member(A -> B, L),  
  del(A -> B, L, NewL),  
  nl,  
  write('\tFirst branch: '),  
  nl,  
  write('= \t'),  
  write([B | NewL] => R),  
  write('\t (by arrow/left)'),  
  prove([B | NewL] => R),  
  nl, You, 1 second ago • Uncommitted changes  
  write('\tSecond branch: '),  
  nl,  
  write('= \t'),  
  write(NewL => [A | R]),  
  write('\t (by arrow/left)'),  
  prove(NewL => [A | R]).
```

Project 3: Your Tasks

- Rule-3: Non-branching rule
- Rule-4: Branching rule